MULTIMEDIA UNIVERSITY ™

STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 2, 2019/2020

DMT5018 – MICROCONTROLLER TECHNOLOGY

(Diploma in Electronic Engineering)

11 MARCH 2020
09:00 AM – 11:00 AM
(2 Hours)

INSTRUCTIONS TO STUDENT

1. This booklet consists of 8 pages including cover pages with 4 questions only.
2. Attempt **ALL** questions given. All questions carry equal marks and distribution of the marks for each question is given.
3. Please write all your answers in the Answer Booklet provided.
4. All necessary working **MUST** be shown.
5. Opcode map and Special Function Register formats are provided in Appendixes.

QUESTION 1 [25 Marks]

(a) With the aid of a suitable diagram, describe what are the major components in a computer. Briefly discuss the function of each component. [10 marks]

(b) Analyze the following sequence of instructions:

```
        ORG  0000H
MAIN:   MOV  24H, #25H
        MOV  25H, #24
        SETB 24H
        CPL  25H
        MOV  A, #25H
        END
```

(i) Determine the content of byte address 24H, and 25H after the program execution. [4 marks]

(ii) Find the content of the accumulator (A) and the program status words (PSW). [1 mark]

(c) Design an 8051-microcontroller capable to address a contiguous of 64 Kbytes of external memory. The lower 32 Kbytes are for program/code and the upper 32 Kbytes are for data. Given the available memory ICs are 16 Kbytes ROM and 16 Kbytes RAM. Write the starting and ending address from each memory block. (Hint: Use decoder where necessary)

[10 marks]

QUESTION 2 [25 Marks]

(a) Briefly explain THREE different addressing modes of 8051 with example for each. [9 marks]

(b) Write a subroutine which checks the contents of byte address 22H. If it is a positive number, the subroutine will find its two's complement and store it in same location and returns. [5 marks]

(c) If the crystal frequency (XTAL) of 8051 is 12MHz, find the time take to execute the following program: [5 marks]

```
        MOV  R3, #04
        MOV  R4, #06
WAIT:   DJNZ R3, WAIT
```

Continued ...

(d) Table Q2 illustrates the program counter (PC) with its corresponding machine codes (in hexadecimal) for the following 8051 programs:

```
        ORG  0099H
        MOV  R5, #25H
READ:   MOV  A, P1
        ANL  A, #1
        JZ   READ
        MOV  P0, P1
```

Table Q2

Program Counter	Machine Codes (H)
	7D 25
009B	

Solve the following questions:

(i) Complete the Table Q2 by filling out the empty spaces. [4 marks]

(ii) Describe the content of the accumulator (A), P0 and P1 after executing the instructions, given the input to Port 1 (P1) is FFH. [2 marks]

QUESTION 3 [25 Marks]

(a) With P0.0 is being defined as an input, and P1.0 and P1.1 are being as outputs, use MCS-51 instruction sets to create the following assembly language program:

(i) A subroutine named "OUTPUT_ON" that set P1.0 and P1.1. [2 marks]

(ii) A subroutine named "OUTPUT_OFF" that clear P1.0 and P1.1. [2 marks]

(iii) A complete program that continuously read P0.0 and call subroutine (i) if P0.0 is HIGH, and call subroutine (ii) if P0.0 is LOW. [5 marks]

Continued ...

(b) With crystal frequency, XTAL = 11.0592 MHz, find the timer value needed to have the following baud rates:

(i) 9600 bps [3 marks]

(ii) 1200 bps [3 marks]

(c) Write a program to transmit "Y", "E", and "S" continuously through a serial port at 4800 baud rate, using 8-bit ASCII code, along with one stop bit. Use Timer1 to provide the clock for the required baud rate. (*Assume 11.0592MHz crystal frequency, SMOD = 1).* [10 marks]

QUESTION 4 [25 Marks]

(a) Assume that after reset, the interrupt priority is set with the instruction **MOV IP, #00001100B**. Discuss the sequence in which the interrupts are serviced. [4 marks]

(b) Write a program that continuously get 8-bit data from P0 and sends it to P2 while simultaneously creating a square wave of 200 μs period on pin P1.1. Use Timer 0 to create the square wave. Assume that XTAL = 11.0592 MHz. [12 marks]

(c) **Figure Q5** shows an electronic door system which will generate a tone for 1 second (using a loudspeaker connected to P2.1) whenever a door sensor connected to an interrupt pin ($\overline{INT0}$) makes a high-to-low transition. Write a program to perform the above mentioned tasks. [9 marks]

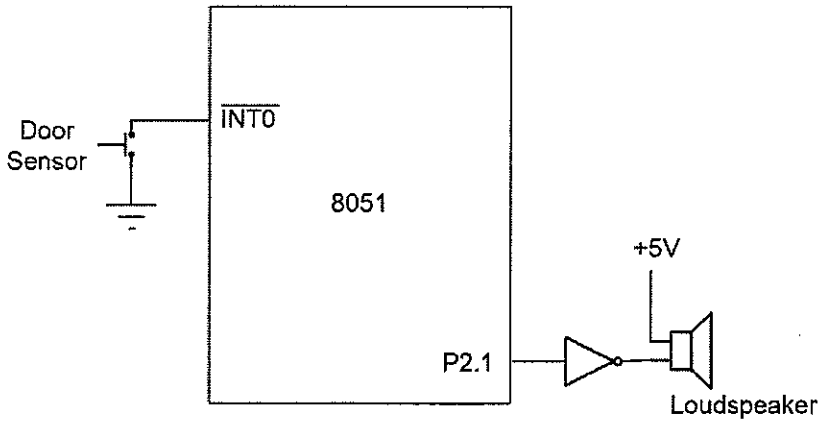
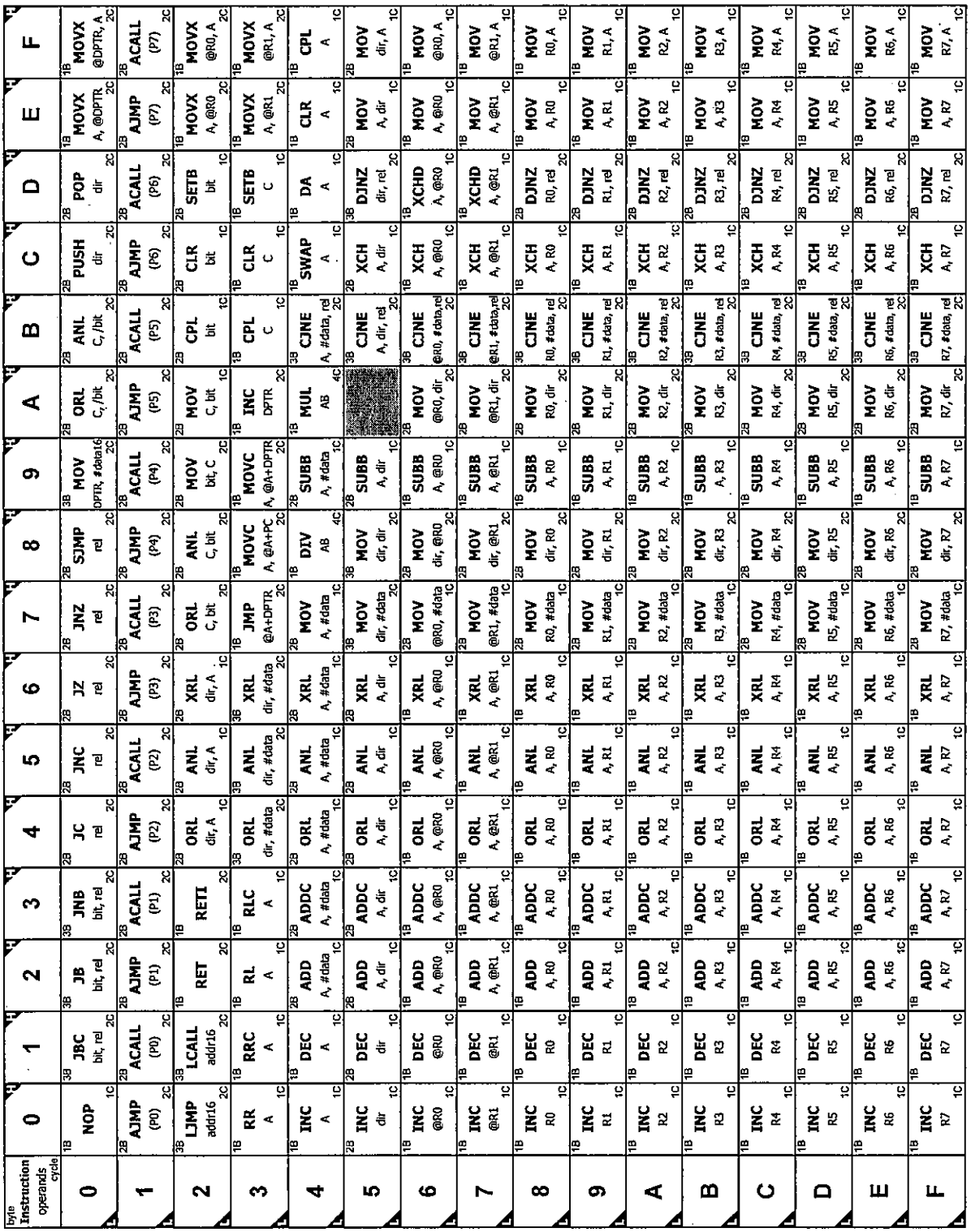


Figure Q5

Continued ...

Appendix A: Opcode Map


byte
Instruction
operands
cycle
0 NOP 1C | JBC bit, rel 2C | JB bit, rel 2C | JNB bit, rel 2C | JC rel 2C | JNC rel 2C | JZ rel 2C | JNZ rel 2C | SJMP rel 2C | MOV DPTR, #data16 2C | ORL C, /bit 2C | ANL C, /bit 2C | PUSH dir 2C | POP dir 2C | MOVX A, @DPTR 2C | MOVX @DPTR, A 2C
1 AJMP (P0) 2C | ACALL (P0) 2C | AJMP (P1) 2C | ACALL (P1) 2C | AJMP (P2) 2C | ACALL (P2) 2C | AJMP (P3) 2C | ACALL (P3) 2C | AJMP (P4) 2C | ACALL (P4) 2C | AJMP (P5) 2C | ACALL (P5) 2C | AJMP (P6) 2C | ACALL (P6) 2C | AJMP (P7) 2C | ACALL (P7) 2C
2 LJMP addr16 2C | LCALL addr16 2C | RET 2C | RETI 2C | ORL dir, A 1C | ORL dir, #data 2C | XRL dir, A 1C | XRL dir, #data 2C | ORL C, bit 2C | JMP @A+DPTR 2C | MOV bit, C 2C | MOVC A, @A+DPTR 2C | CPL bit 1C | CPL C 1C | CLR bit 1C | CLR C 1C | SETB bit 1C | SETB C 1C | MOVX A, @R0 2C | MOVX @R0, A 2C
3 RR A 1C | RRC A 1C | RL A 1C | RLC A 1C | ORL A, #data 2C | ORL A, dir 1C | XRL A, #data 2C | XRL A, dir 1C | ANL C, bit 2C | MOVC A, @A+PC 2C | DIV AB 4C | MUL AB 4C | CPL A 1C | SWAP A 1C | DA A 1C | CLR A 1C | MOV A, dir 2C | MOV dir, A 1C
4 INC A 1C | DEC A 1C | ADD A, #data 2C | ADDC A, #data 2C | ORL A, #data 2C | ANL A, #data 2C | XRL A, #data 2C | MOV A, #data 2C | DIV AB 4C | SUBB A, #data 2C | CJNE A, #data, rel 3C | [redacted] | XCH A, dir 2C | XCH A, @R0 1C | XCHD A, @R0 1C | MOV A, @R0 1C | MOV @R0, A 1C
5 INC dir 1C | DEC dir 1C | ADD A, dir 2C | ADDC A, dir 2C | ORL A, dir 2C | ANL A, dir 2C | XRL A, dir 2C | MOV dir, #data 3C | MOV dir, dir 2C | SUBB A, dir 2C | CJNE A, dir, rel 3C | MOV @R0, dir 2C | XCH A, dir 2C | DJNZ dir, rel 3C | MOV A, dir 2C | MOV dir, A 1C
6 INC @R0 1C | DEC @R0 1C | ADD A, @R0 1C | ADDC A, @R0 1C | ORL A, @R0 1C | ANL A, @R0 1C | XRL A, @R0 1C | MOV @R0, #data 1C | MOV dir, @R0 2C | SUBB A, @R0 1C | CJNE @R0, #data, rel 2C | MOV @R0, dir 2C | XCH A, @R0 1C | XCHD A, @R0 1C | MOV A, @R0 1C | MOV @R0, A 1C
7 INC @R1 1C | DEC @R1 1C | ADD A, @R1 1C | ADDC A, @R1 1C | ORL A, @R1 1C | ANL A, @R1 1C | XRL A, @R1 1C | MOV @R1, #data 1C | MOV dir, @R1 2C | SUBB A, @R1 1C | CJNE @R1, #data, rel 2C | MOV @R1, dir 2C | XCH A, @R1 1C | XCHD A, @R1 1C | MOV A, @R1 1C | MOV @R1, A 1C
8 INC R0 1C | DEC R0 1C | ADD A, R0 1C | ADDC A, R0 1C | ORL A, R0 1C | ANL A, R0 1C | XRL A, R0 1C | MOV R0, #data 1C | MOV dir, R0 2C | SUBB A, R0 1C | CJNE R0, #data, rel 2C | MOV R0, dir 2C | XCH A, R0 1C | DJNZ R0, rel 2C | MOV A, R0 1C | MOV R0, A 1C
9 INC R1 1C | DEC R1 1C | ADD A, R1 1C | ADDC A, R1 1C | ORL A, R1 1C | ANL A, R1 1C | XRL A, R1 1C | MOV R1, #data 1C | MOV dir, R1 2C | SUBB A, R1 1C | CJNE R1, #data, rel 2C | MOV R1, dir 2C | XCH A, R1 1C | DJNZ R1, rel 2C | MOV A, R1 1C | MOV R1, A 1C
A INC R2 1C | DEC R2 1C | ADD A, R2 1C | ADDC A, R2 1C | ORL A, R2 1C | ANL A, R2 1C | XRL A, R2 1C | MOV R2, #data 1C | MOV dir, R2 2C | SUBB A, R2 1C | CJNE R2, #data, rel 2C | MOV R2, dir 2C | XCH A, R2 1C | DJNZ R2, rel 2C | MOV A, R2 1C | MOV R2, A 1C
B INC R3 1C | DEC R3 1C | ADD A, R3 1C | ADDC A, R3 1C | ORL A, R3 1C | ANL A, R3 1C | XRL A, R3 1C | MOV R3, #data 1C | MOV dir, R3 2C | SUBB A, R3 1C | CJNE R3, #data, rel 2C | MOV R3, dir 2C | XCH A, R3 1C | DJNZ R3, rel 2C | MOV A, R3 1C | MOV R3, A 1C
C INC R4 1C | DEC R4 1C | ADD A, R4 1C | ADDC A, R4 1C | ORL A, R4 1C | ANL A, R4 1C | XRL A, R4 1C | MOV R4, #data 1C | MOV dir, R4 2C | SUBB A, R4 1C | CJNE R4, #data, rel 2C | MOV R4, dir 2C | XCH A, R4 1C | DJNZ R4, rel 2C | MOV A, R4 1C | MOV R4, A 1C
D INC R5 1C | DEC R5 1C | ADD A, R5 1C | ADDC A, R5 1C | ORL A, R5 1C | ANL A, R5 1C | XRL A, R5 1C | MOV R5, #data 1C | MOV dir, R5 2C | SUBB A, R5 1C | CJNE R5, #data, rel 2C | MOV R5, dir 2C | XCH A, R5 1C | DJNZ R5, rel 2C | MOV A, R5 1C | MOV R5, A 1C
E INC R6 1C | DEC R6 1C | ADD A, R6 1C | ADDC A, R6 1C | ORL A, R6 1C | ANL A, R6 1C | XRL A, R6 1C | MOV R6, #data 1C | MOV dir, R6 2C | SUBB A, R6 1C | CJNE R6, #data, rel 2C | MOV R6, dir 2C | XCH A, R6 1C | DJNZ R6, rel 2C | MOV A, R6 1C | MOV R6, A 1C
F INC R7 1C | DEC R7 1C | ADD A, R7 1C | ADDC A, R7 1C | ORL A, R7 1C | ANL A, R7 1C | XRL A, R7 1C | MOV R7, #data 1C | MOV dir, R7 2C | SUBB A, R7 1C | CJNE R7, #data, rel 2C | MOV R7, dir 2C | XCH A, R7 1C | DJNZ R7, rel 2C | MOV A, R7 1C | MOV R7, A 1C

Continued ...

Appendix B: Special Function Register (SFR)

Byte address	Bit address								
FF									
F0	F7	F6	F5	F4	F3	F2	F1	F0	B
E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
D0	D7	D6	D5	D4	D3	D2	D1	D0	PSW
B8	-	-	-	BC	BB	BA	B9	B8	IP
B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
A8	AF	-	-	AC	AB	AA	A9	A8	IE
A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
99	not bit addressable								SBUF
98	9F	9E	9D	9C	9B	9A	99	98	SCON
90	97	96	95	94	93	92	91	90	P1
8D	not bit addressable								TH1
8C	not bit addressable								TH0
8B	not bit addressable								TL1
8A	not bit addressable								TL0
89	not bit addressable								TMOD
88	8F	8E	8D	8C	8B	8A	89	88	TCON
87	not bit addressable								PCON
83	not bit addressable								DPH
82	not bit addressable								DPL
81	not bit addressable								SP
80	87	86	85	84	83	82	81	80	P0

Continued ...

Appendix C: Bit-Addressable Ram Locations

Byte Address	Bit Address							
2F	7F	7E	7D	7C	7B	7A	79	78
2E	77	76	75	74	73	72	71	70
2D	6F	6E	6D	6C	6B	6A	69	68
2C	67	66	65	64	63	62	61	60
2B	5F	5E	5D	5C	5B	5A	59	58
2A	57	56	55	54	53	52	51	50
29	4F	4E	4D	4C	4B	4A	49	48
28	47	46	45	44	43	42	41	40
27	3F	3E	3D	3C	3B	3A	39	38
26	37	36	35	34	33	32	31	30
25	2F	2E	2D	2C	2B	2A	29	28
24	27	26	25	24	23	22	21	20
23	1F	1E	1D	1C	1B	1A	19	18
22	17	16	15	14	13	12	11	10
21	0F	0E	0D	0C	0B	0A	09	08
20	07	06	05	04	03	02	01	00

Appendix D: Program Status Word (PSW)

CY	AC	F0	RS1	RS0	OV	--	P

Appendix E: TCON (Timer/Counter Control Register)

TF1	TR1	TF0	TR1	IE1	IT1	IE0	IT0
7	6	5	4	3	2	1	0

Appendix F: TMOD

GATE	C/T	M1	M0	GATE	C/T	M1	M0
7	6	5	4	3	2	1	0

Appendix G: IE (Interrupt Enable Register)

EA	--	ET2	ES	ET1	EX1	ET0	EX0
7	6	5	4	3	2	1	0

Appendix H: IP (Interrupt Priority Register)

--	--	PT2	PS	PT1	PX1	PT0	PX0
7	6	5	4	3	2	1	0

Continued ...

Appendix I: Interrupt Vectors

Interrupt	Flag	Vector Address
Reset	RST	0000H
External 0	IE0	0003H
Timer 0	TF0	000BH
External 1	IE1	0013H
Timer 1	TF1	001BH
Serial Port	RI or TI	0023H

Appendix J: SCON (Serial Port Control Register)

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
7	6	5	4	3	2	1	0

End of Paper